M-1502

Visual Communications and Image Processing (VCIP), January 2001.

Reliable Video Communication over Lossy Packet Networks using Multiple State Encoding and Path Diversity

John G. Apostolopoulos

Hewlett-Packard Laboratories
Palo Alto, CA, USA
japos@hpl.hp.com

ABSTRACT

Video communication over lossy packet networks such as the Internet is hampered by limited bandwidth and packet loss. This paper presents a system for providing reliable video communication over these networks, where the system is composed of two subsystems: (1) multiple state video encoder/decoder and (2) a path diversity transmission system. *Multiple state video coding* combats the problem of error propagation at the decoder by coding the video into multiple independently decodable streams, each with its own prediction process and state. If one stream is lost the other streams can still be decoded to produce usable video, and furthermore, the correctly received streams provide bidirectional (previous and future) information that enables improved state recovery for the corrupted stream. This video coder is a form of multiple description coding (MDC), and its novelty lies in its use of information from the multiple streams to perform state recovery at the decoder. The *path diversity transmission system* explicitly sends different subsets of packets over different paths, as opposed to the default scenarios where the packets proceed along a single path, thereby enabling the end-to-end video application to effectively see an average path behavior. We refer to this as *path diversity.* Generally, seeing this average path behavior provides better performance than seeing the behavior of any individual random path. For example, the probability that all of the multiple paths are simultaneously congested is much less than the probability that a single path is congested. The resulting path diversity provides the multiple state video decoder with an appropriate virtual channel to assist in recovering from lost packets, and can also simplify system design, e.g. FEC design. We propose two architectures for achieving path diversity, and examine the effectiveness of path diversity in communicating video over a lossy packet network.

Keywords: Error-resilient video coding, video streaming, multiple description coding, path diversity, packet video, wireless video

1. INTRODUCTION

Multimedia communication over packet networks such as the Internet is rapidly gaining in importance. Currently, applications such as video and audio communication are hampered by the limited bandwidth and packet loss that afflict the Internet. These applications require high compression and high error resilience, however simultaneously achieving these qualities is difficult because these are largely conflicting requirements.

This paper presents a system for reliable video communication over lossy packet networks such as the Internet. The proposed system is composed of two jointly designed subsystems: (1) multiple state video encoding and decoding, and (2) a path diversity transmission system. Multiple state video coding is designed to combat the error propagation problem that afflicts motion-compensated prediction based coders when there are losses. In this approach the video is coded into multiple independently decodable streams, each with its own prediction process and state. If one stream is lost the other streams can still be decoded to produce usable video, and furthermore, the correctly received streams provide bidirectional (previous and future) information that enables improved state recovery for the corrupted stream. The video coder is a form of multiple description coding (MDC), e.g. Ref. 1, and its novelty lies in its use of information from the multiple streams to perform state recovery at the decoder.

The proposed path diversity transmission system explicitly sends different subsets of packets for an application over different paths, as opposed to the default scenario where the stream of packets proceeds along a single path. By using multiple paths at the same time the end-to-end video application effectively sees an "average" path behavior. We refer to this as path diversity. Generally, seeing this average path behavior provides better performance than seeing the behavior of any randomly chosen individual path. The benefits of path diversity include (1) the application sees a virtual average path which exhibits a smaller variability in communication quality than exists over an individual

path, (2) burst packet losses are converted to isolated packet losses, and (3) the probability of an outage (where all packets in the packet stream are lost for the duration of the outage) is greatly reduced. These improvements provide some interesting benefits to video communication performance under packet loss, and may also simplify general packet-based communication system design. We propose two architectures for explicitly sending a stream of packets through multiple paths, based on (1) IP Source routing and (2) a relay infrastructure. The relay infrastructure appears particularly promising for today's Internet, and corresponds to an application-specific overlay network on top of the conventional Internet. The proposed system routes traffic through semi-intelligent nodes at strategic locations in the Internet, thereby providing a service of improved reliability while leveraging the infrastructure of the Internet.

The proposed system does not rely on a back-channel between the decoder and encoder. Therefore it is applicable to both closed-loop (when a back channel is available) and open-loop (when a back channel is not available) applications. Important open-loop video communication applications include broadcast, multicast, or point-to-point with unreliable backchannel. The use of a back channel limits the scalability of one-to-many systems (e.g. multicast) because of the overhead associated with the many responses. Therefore, it is beneficial for a system to provide reliable video communication without relying on a back channel, even if one is available. The proposed system is also not dependent on the network supporting different qualities of service (e.g. high and low priority packets), therefore its performance is not adversely affected when all packets are equally likely to be lost.

This paper continues in Section 2 by briefly describing the basic problems that afflict compressed video in error-prone environments and the modern approaches developed to overcome these problems. We continue in Section 3 by presenting our multiple state video coding approach for combating the problem of incorrect state and error propagation at the decoder. Section 4 provides background and motivation for our proposed path diversity approach, including a discussion of the problems that afflict packet-based communication and the conventional approaches to overcome packet loss. The proposed path diversity approach is presented in Section 5, along with two architectures for achieving path diversity in different networks, and a discussion of the benefits of path diversity. The proposed multiple state coding and path diversity system is presented in Section 6, followed by some experimental results in Section 7 to illustrate the effectiveness of the proposed approach. This paper concludes with a summary.

2. PREVIOUS WORK ON ERROR-RESILIENT VIDEO CODING

Most video compression systems possess a similar architecture based on motion-compensated (MC) prediction between frames, Block-DCT (or other spatial transform) of the prediction error, followed by entropy coding of the parameters. The basic error-induced problems that afflict a system based on this architecture include:

Bitstream Synchronization With the use of entropy coding (e.g. Huffman coding) an error can cause the decoder to lose synchronization with the bitstream, i.e. the decoder may not know what bits correspond to what parameters. Approaches for overcoming this problem include: placing resynchronization markers at strategic locations in the compressed video hierarchy (e.g. picture or slice headers), placing resync markers after every fixed number of bits (variable number of blocks), organizing the variable length coded blocks so that each block starts at a known location in the bitstream, partitioning the data into groups based on importance, and reversible variable length codes for (partial) recovery of lost data.[2–6]

Incorrect State at Decoder Even if the bitstream has been resynchronized, another crucial problem is that the state of the representation at the decoder may not be the same as the state at the encoder. In particular, when using MC-prediction an error causes the reconstructed frame to be incorrect and often leads to significant error propagation to subsequent frames. We refer to this problem as having incorrect (or mismatched) state at the decoder, because the state of the representation at the decoder (the previous coded frame) is not the same as the state at the encoder. This problem also arises in other contexts (e.g. random access or channel acquisition) and a number of approaches have been proposed to overcome it including: independent coding of each frame (all Intra-frame or I-frame coding), periodic I-frames to periodically reinitialize the prediction loop (e.g. MPEG GOP), leakage within the prediction loop, and partial intra-encoding of each frame. While intra coding limits the effect of errors, the high bit rate required for intra coding limits its use in many applications.

The special case of point-to-point transmission with a back-channel facilitates additional approaches including: the decoder notifying the encoder to (1) reinitialize the prediction loop, or (2) which frames where correctly/erroneously received and therefore which frame should be used as the reference for the next prediction (referred to as NewPred in MPEG-4 Version 2 and as Reference Picture Selection (RPS) in H.263 Version 2[7–9,4,3]). NewPred can be very valuable

in the case of a point-to-point link which also has a reliable back channel and with sufficiently short round-trip-delay; otherwise the visual degradation can be quite significant.[7]

RPS can also be applied without a back channel in an approach referred to as Video Redundancy Coding (VRC)[10] where most frames are assigned to one of two or more independently coded threads and at periodic intervals (e.g. 7 or 10 frames for 2 or 3 threads[10]) a single frame is chosen as a sync frame and is coded redundantly into each of the threads to enable synchronization; the decoder receives multiple copies of each sync frame and decodes one error-free copy while discarding the rest. If one thread is lost because of an error, the next sync frame can be used for recovery. Because VRC codes each sync frame multiple times and because of the reduced prediction accuracy that results from predicting frames spaced further apart in time, there is an extra overhead of approximately 35% and 57% for the 2 and 3 thread cases, respectively.[10] Other open-loop approaches involving multiple reference frames select an appropriate frame for prediction based on both its prediction accuracy and an estimate of its ability to provide additional robustness against error propagation.[11–13]

Layered or scalable approaches essentially prioritize data and thereby support intelligent discarding of the data (the enhancement data can be lost or discarded while still maintaining usable video), however the video can be completely lost if there is an error in the base layer. Multiple Description Coding (MDC) attempts to overcome this problem by coding a signal into multiple bitstreams such that any one bitstream can be used to decode a baseline signal, and any additional bitstreams will improve the quality of the reconstructed signal. Recent application of MDC ideas to video coding are based on predictive MD quantizer,[1] MD transform coding,[14] and multiple states.[15]

Three-dimensional subband coding has been examined as an alternative to conventional MC-prediction based schemes, and it provides interesting benefits for error-resilient video communication. In particular, Ref. 16 proposes an approach where every packet is independently decodable and has approximately equal expected visual importance. This is achieved by appropriately interleaving the 3-D subbands among the packets, and since there is no prediction loop there is no error propagation. The primary drawback of a 3-D subband type approach is that the additional delay required for the temporal subband decomposition usually precludes its application for real-time video communication.

This section discussed the two major classes of problems that afflict compressed video communication in error-prone environments: (1) bitstream synchronization and (2) incorrect state and error propagation. The problem of bitstream synchronization can often be largely minimized through appropriate algorithm and system design. Specifically, for video communication over a packet network the video encoder may know the packet size and can design the packet payloads so that each packet is independently decodable, i.e. bitstream resynchronization is supported at the packet level so that the various coding parameters in each correctly received packet can be straightforwardly parsed and decoded. Both MPEG-4 and H.263+ support the creation of different forms of independently decodable "video packets". This is an example of the Application Level Framing principle,[17] which essentially says that the application (in this case the video coder) knows best how it can handle packet loss, out-of-order delivery, and delay, and therefore the application should design the packet payloads. The use of this principle largely overcomes the bitstream synchronization problem in video communication over lossy packet networks.

A variety of algorithms have been proposed for error concealment and control,[18] however the problem of incorrect state and error propagation at the decoder remains a major obstacle to reliable video communication over packet networks such as the Internet*. Therefore, the goal of this work is to design a video coding and communication system that can overcome the problem of incorrect state and error propagation at the decoder. Specifically, we assume no back-channel between the decoder and encoder (e.g. broadcast, multicast, or point-to-point with unreliable backchannel) and that it is not possible to specify different qualities of service (i.e. all packets are equally likely to be lost).

3. PROPOSED MULTIPLE STATE VIDEO CODING

Conventional video compression standards employ a similar architecture which we refer to as single-state systems since they have a single state (e.g. the previous coded frame) which if lost or corrupted can lead to the loss or severe degradation of all subsequent frames until the state is reinitialized (the prediction is refreshed). In our proposed

*In certain special cases, such as a point-to-point link with a backchannel and with sufficiently short and reliable round-trip-delay (RTD), NewPred/ReferencePictureSelection can overcome the problem of incorrect state at the decoder. However, many important applications do not have a backchannel, and in other applications the backchannel may be unreliable or has a long RTD, thereby severely limiting NewPred's effectiveness.

approach we code the video into a number of independently decodable streams, each with its own prediction process and state information, as shown in Figure 1.[15] By having multiple (independently decodable) state streams, if one state is corrupted the other states remain accurate and their respective streams can still be accurately decoded to produce usable video and may also be used to recover the lost state. In particular, the novelty of our approach lies in our using of data from the multiple streams to recover the lost state. Specifically, we exploit the redundancy between frames in the different streams to improve the recovery of the lost frames.

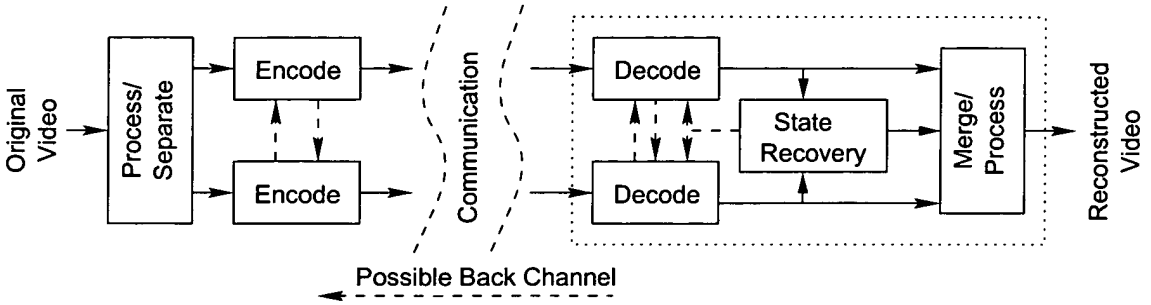

Figure 1. A general two-state video communication system.

3.1. Encoder Portion of System

In the simplest instance of the proposed approach, the input video is partitioned into two subsequences of frames (even and odd) which are coded into two separate bitstreams. Specifically, each stream has a different prediction loop and a different state, and is independently decodable from the other. Since in general there can be multiple coded streams each with its own state we refer to this approach as *Multiple State Encoding.*

The encoder may consist of two separate conventional encoders, or an encoder which stores the last two previously coded frames (instead of just the last one) and chooses which previously coded frame to use to form the prediction for the current frame to be encoded. Both MPEG-4 and H.263+ support switching prediction among reference frames.

A higher bit rate is required to code the frames in separate subsequences as opposed to a single sequence, since they are spaced farther apart in time and prediction does not perform as well. However, unlike video redundancy coding there are no redundantly coded frames. The proposed approach is conceptually similar to multiple description coding, e.g. Ref. 1, however it differs in the representation used for each description and most importantly in its use of state recovery.

The different streams should be transmitted over different channels undergoing independent error effects to minimize the chance that both streams are lost. For example, the bitstreams from the even and odd frames can be sent in different packets over a packet network, so that any lost packet will only affect one of the streams. This important issue is examined in more depth in our proposed system in Section 6.

3.2. Decoder Portion of System

In a manner similar to the encoder, the decoder may consist of two separate decoders, or a single decoder that alternates which previous decoded frame it uses to perform the prediction. If there are no errors and both the even and odd streams are received correctly, then both streams are decoded to produce the even and odd frames which are interleaved for final display.

If a stream has an error then the state for that stream is incorrect and there will be error propagation for that stream. However, the other independently decodable stream can still be accurately and straightforwardly decoded to produce usable video. For example, if the bitstream corresponding to the odd frames is lost, the even frames may still be decoded and displayed, recovering the video at half its original frame rate. The error produces a temporary reduction in the frame rate, however there are no other distortions — which may be preferable to the case of conventional (single-state) approaches which are forced to either freeze the video or attempt to estimate the unknown video by performing some form of concealment; either of which can lead to significant distortion, especially if there are many frames before the next I-frame. The drawback of this simple approach is that if there are multiple

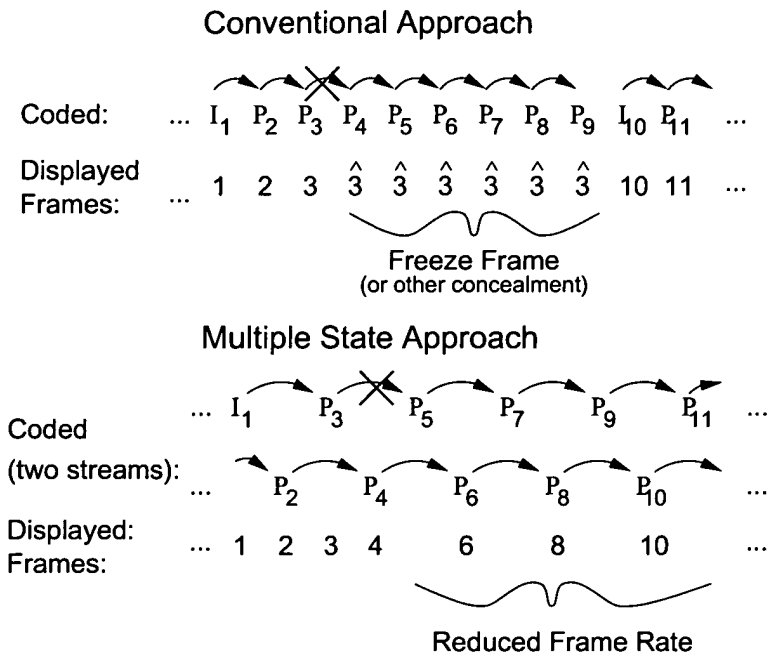

Figure 2. The effects of an error in decoding the frame that depends on frame P_3: In a conventional single-state approach (top) frame 4 is lost and the decoder may freeze frame 3 (or perform other error concealment) until the next I-frame. In a simple two-state stream approach (bottom) stream #1 is lost, however stream #2 can be accurately decoded – recovering the video at half its original frame rate but without any other distortions. More importantly, stream #1 can often be recovered by appropriately using stream #2.

errors before the next I-frame then both streams may be affected and the situation would be similar to that of the single-state approach.

The novelty in the multiple state approach is that it provides improved error concealment and enables improved state recovery of the corrupted stream. Conventional single-state approaches only have access to *previous* frames to use in error concealment. The proposed approach provides access to both *previous and future frames*, as illustrated in Figure 3. The availability and careful usage of both previous and future frames can greatly assist in recovering the corrupted stream and thereby restore the video to its full frame rate. Specifically, the lost state (the coded frame) can often be estimated with sufficient accuracy to be used as a reference for predicting other frames in that stream. As a result, the corrupted stream may be recovered quickly, which is preferable to waiting for the next resynchronization.

In contrast to the conventional single-state architecture, which provides access to only previous frames to perform the concealment (or state recovery), the proposed approach provides access to both previous and future frames, enabling improved state recovery. In addition, the use of multiple states provides the capability to estimate the *quality of the state recovery*. For example, in a manner analogous to how the correctly received stream can be used to estimate the corrupted stream, the recovered corrupted stream can be used to estimate the known correctly received stream, and the accuracy of the match can provide an estimate of the recovery quality. Note that in the conventional single-state approach it is typically very difficult for the decoder to estimate the quality of the resulting error concealment since the decoder has no knowledge of what the correct frames should be. Knowledge of the quality of the error concealment may be beneficial in a variety of ways, e.g. if the quality is unacceptable then the decoder may choose to simply freeze the last correctly decoded frame and wait for the next resync, while if the quality is good it can continue to decode and display all the frames.

3.3. State Recovery By Using Multiple States

The problem of state recovery is similar to that of MC-interpolation (MC-I) where a frame is estimated using both previous and future frames in the sequence. MC-I has received considerable attention over the years and many of

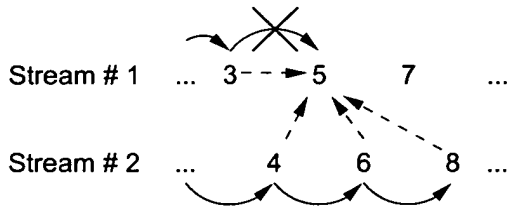

Figure 3. Correctly received streams enable improved error concealment and state recovery for the corrupted stream. Frame 5 may be concealed/recovered by using information from previous and *future* correctly decoded frames as signified by the dashed lines.

the algorithms and results developed for MC-I can be used in our context for state recovery. However, while there are similarities between the problems of state recovery and MC-I, there are also differences. First, state recovery and MC-I have subtlely different goals. MC-I is explicitly designed to produce enhanced-quality video for display and thus tries to construct frames that are visually very pleasing. The goal of state recovery is to produce an accurate estimate of the state (coded frame) so that it can be used to form an accurate prediction of the subsequent frames. Therefore, prediction accuracy, not visual quality, is the most important criterion for state recovery. Second, state recovery attempts to recover a coded (distorted) frame from previous and future coded (distorted) frames. This contrasts with MC-I which tries to interpolate a clean frame from other clean frames. This has a number of implications, e.g. the conventional MC-I approach of estimating the motion between the previous and future frames may often be inappropriate. Third, it is often desirable for the decoder to perform state recovery in real time with low complexity. And fourth, the decoder has access to coded motion vectors and other information that may be useful for performing the state recovery.

There are a variety of possible approaches for estimating the lost frame. These include low-complexity approaches such as simply replacing the lost frame by a correctly decoded frame, or a MC correctly decoded frame, or a more sophisticated MC-I algorithm, e.g. compute the motion field across a subset of correctly decoded past and future frames from the corrupted and uncorrupted streams, and apply appropriate linear or nonlinear filtering along the motion trajectories. The recovery (interpolation) should also account for covered and uncovered areas within the frame by appropriately choosing to use only future or previous frames to estimate the appropriate areas. An adaptive method which selects the appropriate recovery method as well as the appropriate subset of past and future frames based on the specific video context would be most effective. The coded information within the bitstreams (e.g. motion vectors, inter/intra decisions) can be used instead of, or in addition to, performing motion estimation on the coded frames. The use of coded information can significantly reduce the complexity of state recovery at the decoder and may in certain cases improve its effectiveness.

4. OVERVIEW OF PACKET-BASED COMMUNICATION

This section provides a brief overview of conventional communication over packet networks. The discussion is short and highly simplified, and is included solely to provide adequate background on current approaches for appropriate comparison versus the proposed path diversity approach. More thorough discussions of conventional packet-based communication is available in a variety of sources, including Ref. 19. This section continues by roughly classifying the types of packet loss that may occur and describes how they affect a video communication system.

4.1. Problems that Afflict Packet-based Communications

In conventional packet-based communication, such as over the Internet, the sender drops packets on the network with the destination IP address, and these packets are delivered to that address. This process is analogous to how postal mail is delivered from sender to receiver. The sending and receiving computers have no control over how the packets get from point A (sender) to point B (receiver). This property has been a great asset to the growth of the Internet, e.g. the infrastructure can be completely changed without affecting how point A communicates with point B.

The problems of packet-based communication from an application viewpoint (end-to-end viewpoint) include packet loss, variable delay, and out-of-order delivery. In addition, a network typically exhibits dynamic (time-varying) characteristics, and the application usually has minimal knowledge of network characteristics.

The effect of packet loss is highly application dependent. For example, applications such as Web traffic, FTP, or telnet, are point-to-point and a back-channel is available and therefore packet loss may be overcome by simply notifying the sender of the lost packets and retransmitting those packets. This approach leads to a delay, however in the applications described above this delay is acceptable. However, in a number of applications the use of retransmissions may not be possible because of (1) delay constraints (data has time-bounded usefulness), and (2) lack of a feedback channel to request the retransmits or inability to use retransmits (e.g. in a broadcast or multicast situation where retransmit requests lead to the implosion problem). Also, in certain applications a back channel may be available but it is desirable not to use the back channel in order to produce a more scalable system. Important applications with these constraints include video and audio communication, such as two-way real-time video phone or video conferencing (which have delay constraints), interactive one-way video such as video games (which has delay constraints), and one-way broadcast or multicast video or audio to a large number of receivers (where retransmits are generally not possible or undesirable). In these and other applications it is desirable or necessary to have *reliable feedback-free communication* – the application should run reliably even when there is packet loss.

The effect of packet loss greatly depend on the type of loss and the particular application. The types of packet loss can be roughly grouped into the following three classes[†]:

1. Isolated, single-packet loss
2. Burst loss: multiple (roughly consecutive) packets lost
3. Temporary outage: complete (temporary) loss of communication, e.g. .5 seconds to several seconds

The effect of the different types of packet loss depends critically on the particular application. When communicating compressed video, the video is compressed and packetized into independently decodable packets (e.g. video packets in MPEG-4 terminology). While this approach overcomes the problem of bitstream synchronization, the problem of error propagation remains. As described in Section 2, the contents of each packet are dependent on the contents of other (usually previous) packets to reconstruct the video and the loss of a single packet affects the use of other (correctly received) packets. Therefore, the propagation effect of a loss can be quite substantial.

On the other hand, video has substantial spatial and temporal correlations which may be used to estimate the lost information. Specifically, the loss of a small amount of information, such as a portion of a frame, may be concealed through the use of sophisticated error concealment techniques. However, if a large amount of information is lost, for example multiple consecutive frames, then the effect is much more detrimental. The effect of packet loss depends on the type of loss. For communication of QCIF or CIF video over the Internet, each frame is typically compressed into a small number of packets, e.g. one to five packets/frame[‡]. Therefore, for compressed video, the different types of packet losses described above lead to the following rough but useful observations:

1. Isolated, single-packet loss corresponds to the loss of one frame or a portion of one frame; the video data may be partially recoverable.
2. Burst-loss corresponds to one or a number of frames being lost, which may lead to significant visual degradation.
3. Outage results in a number of frames being lost, which typically results in a total loss of the video; the system can not recover without an I-frame for resynchronization.

It is important to note that the loss of a number of consecutive packets has a much more detrimental effect than the loss of a single packet, and often more than the loss of an equivalent number of isolated single packets. This is because the loss of a small amount of video can often be recovered by exploiting the spatial and temporal correlations that exist between the video that was lost and that which was received, however the effectiveness of this recovery decreases rapidly as a larger amount of contiguous video is lost. Therefore, from a video communication perspective, *it is important to reduce/eliminate the effects of burst losses and outages.* These observations were partial motivations for the system proposed in Section 6.

[†]In addition, infrequently a system may go down leading to a long-term outage lasting multiple hours or days.

[‡]For example, the QCIF and CIF compressed video sequences discussed in Section 7 result in approximately one and five (approx) 1500 byte packets/P-frame, respectively.

4.2. Conventional Approaches to Overcome Packet Loss

The conventional approaches to overcome packet loss include retransmission and forward error correction (FEC).

Retransmission-based approaches use a back-channel to enable the receiver to communicate to the sender which packets were correctly received and which were not. Retransmission leads to additional delay corresponding roughly to the round-trip-time (RTT) between receiver-sender-receiver. In many applications this delay is acceptable, e.g. Web browsing, FTP, telnet. For example, when guaranteed delivery is required (and a backchannel is available) then feedback-based retransmits enable an application to adapt to dynamic channel conditions and use no more resources than required. On the other hand, in cases when a back-channel is not available or when delay is not acceptable, then retransmission is not an appropriate solution.

FEC-based approaches add specialized inter-packet redundancy (e.g. Reed Solomon block codes and Tornado codes) to the data to overcome losses. FEC approaches may also interleave the packets to convert burst errors into isolated errors. The redundancy added in FEC-based approaches leads to increased bandwidth requirements. The FEC-based approaches are designed to overcome a predetermined amount of channel losses. If the losses are less than the threshold, then the transmitted data can be perfectly recovered from the received, lossy data. However, if the losses are greater than the threshold, then only a portion of the data may be recovered, and depending on the type of FEC used, the data may be completely lost.

In most network applications, the network conditions such as packet loss are highly dynamic and there is limited knowledge about the current conditions (time scales for changes are shorter than measurement time scales). The lack of knowledge about the instantaneous channel conditions typically leads to inefficient FEC design. Specifically, if the channel is better than that designed for then resources are being wasted (more redundancy than necessary is used). However, if the channel is worse than that designed for then all the data may be lost (not enough redundancy is employed). Because of the highly dynamic nature of many networks, in most cases the FEC is either over-designed and therefore inefficient, or under-designed and therefore ineffective. This is unlike the textbook deep-space communication channel where the channel conditions are static and known and therefore highly effective and efficient channel coding schemes can be designed.

In summary, retransmission-based approaches are not applicable in applications when a back-channel is not available or when the retransmit delay is not acceptable, e.g. broadcast or multicast video and real-time video communication, respectively. FEC-based approaches are also often unsatisfactory because the highly dynamic and unknown network conditions often result in the FEC being either over-designed (and therefore inefficient) or under-designed (and therefore ineffective)[§].

In recent years many sophisticated video compression and communication systems have been designed for communication over a lossy packet network, based on scalable coding, prioritized data, combinations of ARQ and FEC, unequal error protection, multiple description coding, etc.[1,10,18,16,20,21] These proposals provide different advantages and disadvantages and vary in the circumstances where they are usable (e.g. open-loop or closed-loop communication). However, we are not aware of any proposals for video communication over a lossy packet network based on the use of path diversity as proposed in the following sections.

5. PATH DIVERSITY FOR IMPROVING COMMUNICATION OVER LOSSY PACKET NETWORKS

The goal of this work is to determine a more effective and efficient method for video communication over lossy packet networks. By effective we mean reliable (in the application sense) even when packet loss occurs, and by efficient we mean with a minimum of excess as compared to the case where the system is designed for communication when there is no loss. An underlying assumption is that burst losses and outages are especially harmful to video communication. In addition, the highly dynamic nature of packet networks such as the Internet make optimized design very complex. This section continues by describing the general idea of path diversity, and proposing two architectures for enabling path diversity. Some of the benefits of path diversity are then briefly identified and discussed.

In the conventional Internet scenario, a computer drops packets on the network with a destination IP address and the packets are delivered to that destination address. The sending and receiving computers have no control over how

[§]A recent proposal for designing efficient and effective FEC-based schemes for the difficult multicast environment is based on receiver-driven FEC-subscription, where the receiver uses his knowledge of his network conditions to request the appropriate amount of FEC protection.[20]

the packets get from point A (sender) to point B (receiver). An important observation is that while one node or path in the network may be congested, other nodes or paths may have ample bandwidth. It would be advantageous to know the instantaneous quality of each path and to use that information to send packets along paths with available bandwidth (much like listening to a traffic report before leaving for work). However this is very difficult for a number of reasons, including the fact that the congested areas can vary quite rapidly. While it may not be possible to know which paths are the best to use at any point in time, by using a number of paths at the same time some amount of averaging occurs and the application can effectively see the "average" path behavior. As is discussed further in Section 5.3, seeing this average path behavior is generally better than seeing the behavior of any individual path. We refer to this as path diversity.

Diversity techniques have been studied for many years in the context of wireless communication, e.g. frequency, time, and spatial diversity.[22] However to the author's knowledge, the problem of path diversity over a wired packet network has been largely unexplored. A number of thorough studies have shown that there exists great variability in the end-to-end performance observed over the Internet, e.g. Ref. 23. This variability is analogous to the variability that exists in a wireless link and motivated the use of diversity in wireless communications. Therefore, diversity would also appear to be beneficial for communication over the Internet.

The recent work by Savage, Collins, and Hoffman[24] adds justification to our proposal for path diversity. In their measurement-based study, they compared the performance of the default path between two hosts on the Internet to that of alternative paths between those two hosts. They find that "in 30-80% of the cases, there is an alternate path with significantly superior quality", where quality is measured in terms of metrics such as round-trip-time, loss rate, and bandwidth.

5.1. Overview of Path Diversity and Important Issues

Path diversity can be achieved in the following manner. Given a stream of packets to deliver from point A to point B, the different packets are explicitly sent over different paths in order to achieve the benefits of path diversity. Consider a simple example where there are two paths connecting sender A and receiver B, A sends half the packets over one path and the other half of the packets over the other path. Note that in the ideal case the total number of packets transmitted remains constant. In certain applications it may be required or beneficial to send a larger total number of packets over the multiple paths than what would be sent over a conventional single path.

In order to achieve and effectively use path diversity a number of important issues must be addressed including: how to send packets through different paths, how to judge the degree to which two paths are different, how many paths to use, how to choose the paths, how to share the load among the chosen paths, and how to use (potential) feedback from the receiver to improve the communication. These are important and difficult questions, and many are beyond the scope of this paper. In the remainder of this paper we (1) propose two methods for sending packets through different paths (thereby achieving path diversity), (2) propose a system that couples multiple state video coding with path diversity and examine the problem of the number of paths to use, and (3) present some experimental results showing the potential benefits of the proposed system.

5.2. Architectures for Achieving Path Diversity

The path taken by a packet as it traverses a network depends on where the packet enters the network, its destination address, and the state of each of the routing tables for all of the routers that are traversed. Each intermediate router makes a decision about which next-hop router to send the packet to. Typically all of this is transparent to the application, the application just drops the packet to the network layer and the network routes the packet to the desired destination.

As an important aside, consecutive packets within a stream of packets between a sender and receiver generally go through the same path. This is because the time rate of change of routing tables is slow (as compared to the time between packets). However, this is not true if, for example, some sort of load balancing is used. While consecutive packets generally go through the same path, they may see very different delays and loss characteristics. This is because the delays and loss characteristics at a particular node depend on the congestion at that node, which depends on the cross traffic which can vary quite quickly.

The following subsections discuss two different methods for explicitly sending packets through different paths, in roughly increasing order of practicality for the Internet today.

5.2.1. Path Diversity via IP Source Routing

In certain circumstances it is possible to explicitly specify the set of nodes or "source route" for each packet to transverse. One can specify a subset of the nodes (loose source routing) or the complete set of nodes (strict source routing). This approach is referred to as IP Source Routing. Path diversity can be achieved by using IP Source Routing to explicitly specify different source routes for different subsets of packets. Note that IP Source Routing is not novel, the novelty lies in using IP Source Routing to provide path diversity by routing different subsets of packets (within a packet stream) through different paths in a network to the final destination.

The use of IP Source Routing to provide path diversity appears relatively straightforward, however it has a number of problems. First, if the network allows any end-user to specify any source routes it can lead to security problems. Therefore, IP Source Routing is usually turned off within the Internet for security reasons. On the other hand, it may be possible to use IP Source Routing within a private network, for example within the network of a large company. Second, there is the question of how to appropriately choose the paths. This requires knowledge of the network topology and the network state, which is complicated for a number of reasons. For example, in order to specify a route through the Internet (a sequence of node addresses), one needs to know node addresses within the various subnetworks that must be crossed. However, network operators generally do not want outsiders to know the internals of their network. Furthermore, the topology of the Internet is ever changing and it probably would be unadvisable to design an algorithm that requires detailed knowledge of the network topology. In addition, it is probably impractical to know the network state for a large part of the Internet (though it may be possible for an internal company network which is smaller and under complete control, or between a controlled network of nodes within the Internet). Another issue, of lesser importance, is that more overhead is required for carrying the routing information since the intermediate node addresses must be included in each packet. The overhead is equal to the number of specified intermediate nodes times 32 bits/address for IPv4 or 128 bits/address for IPv6.

Based on the above, IP source routing is probably not appropriate for use over the Internet, however it may be useful within a company intranet. As an aside, load balancing is often used within a company intranet, and path diversity (such as enabled by IP source routing) also provides some amount of load balancing. Therefore, there are multiple motivations to use path diversity within a company intranet, as it may provide both improved reliability and load balancing.

5.2.2. Path Diversity via Relays

Another approach to explicitly send packets over different paths is through the use of relays. The relays are placed in a number of important nodes in the infrastructure and each relay performs a simple forwarding operation. The basic idea is to encapsulate an original packet (destination address and payload) into a new packet which is sent to (has the address of) a relay. The relay receives the packet, simply strips of its address, and drops the original packet back on the network where it goes off to the final destination.

For example, consider the scenario in which sender A wants to transmit a stream of packets to receiver B, and a relay C is available. Sender A can send half the packets (e.g. the even numbered packets) directly to B by dropping them on the network as usual. Sender A can then send the other half of the packets (e.g. the odd numbered packets) to B via an alternate path by encapsulating each of them in a new packet and sending them to relay C, which then forwards those packets to receiver B. If two relays were available, sender A could send half of its packets through the first relay and half through the second relay; or alternatively, the sender could send one-third of the packets through the first relay, one-third of the packets through the second relay, and simply drop the remaining one-third of the packets on the network addressed to C and hope that they go through a different path. In certain instances it may be necessary to use separate relays to ensure separate paths and in other cases it may not be necessary. Figure 4 illustrates the use of three relays to specify three different paths between sender and receiver.

In the above example the relay performs relatively simple processing and should be of low complexity. Alternatively, the relay may examine the destination address and perform some appropriate processing, such as reencapsulating it and sending it to another relay close to the final destination. The relay architecture provides considerable freedom for optimization of the relay network. Most of this freedom may be hidden from the clients (applications) which only need to be able to connect to the relay network.

There are a number of ways in which a client or application can use or interact with a relay network to provide path diversity. For example, if a client knows an appropriate relay's IP address then it may communicate directly with (through) the relay. Alternatively, if a relay network architecture or service is established, then the client may simply send its packets to the service and the service automatically provides the relaying and ensures reliable delivery.

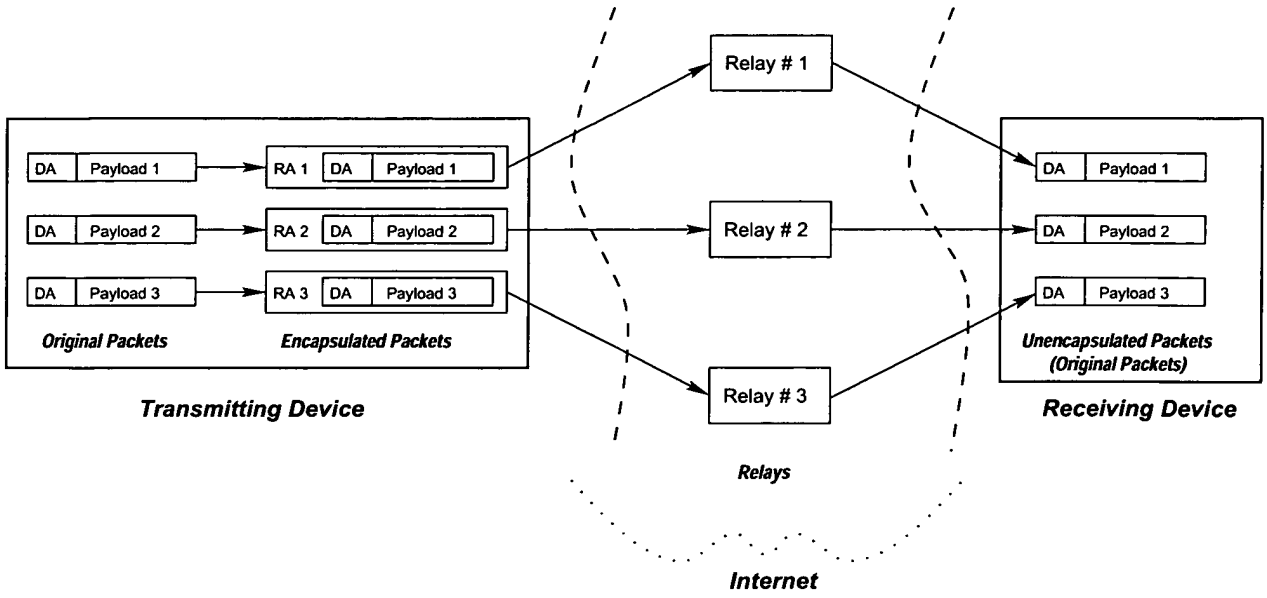

Figure 4. Path diversity through the use of a relay network. In this case, the transmitter sends a stream of packets over the Internet to a receiver using a relay network consisting of three relays. The transmitter partitions the packet stream into three subsets of packets, e.g. packets {1,4,7,10,...}, packets {2,5,8,11,...}, and packets {3,6,9,12,...}, and each stream is sent through a different relay. Specifically, each of the original packets, both destination address (*DA*) and payload, is encapsulated in another packet and sent to the appropriate relay address, e.g. *RA 1*. Each relay peels off its own address (the packet header) from the received packets, and drops the contents (corresponding to the original packets) back on the network for delivery to the final destination.

5.2.3. Summary of Approaches to Explicitly Use Different Paths

The use of IP Source Routing to send different packets over different paths provides a straightforward method to achieve path diversity. While theoretically it may be the simplest approach, practically it is currently not possible in the Internet because source routing is typically turned off in the Internet because of security reasons. However source routing may be possible within a company intranet.

Relay-based path diversity provides another relatively simple method for achieving path diversity, however it does require an infrastructure of relays. However, this approach also enables the establishment of a relay architecture and service for reliable communication over the Internet.

5.3. Benefits of Path Diversity

A number of benefits result from explicitly sending a packet stream over multiple paths, including[25]:

- The end-to-end application sees the average network behavior. This generally leads to a reduction in the variability in link/application quality. A reduced variance in the network characteristics can lead to easier system design and improved end-to-end application quality. For example, reduced variance can enable improved efficiency and effectiveness in an FEC-based system.
- Burst loss (the loss of many consecutive packets) can be converted into the loss of a number of isolated packets. This may provide a number of benefits. For example, in compressed video communication it is easier to recover from multiple isolated losses than from a number of consecutive losses.
- The probability of an outage decreases dramatically with path diversity, because when using path diversity an outage occurs if and only if all paths undergo outages simultaneously. Throughout the remainder of this paper we assume that the loss processes of each path can be modeled as independent and identically distributed

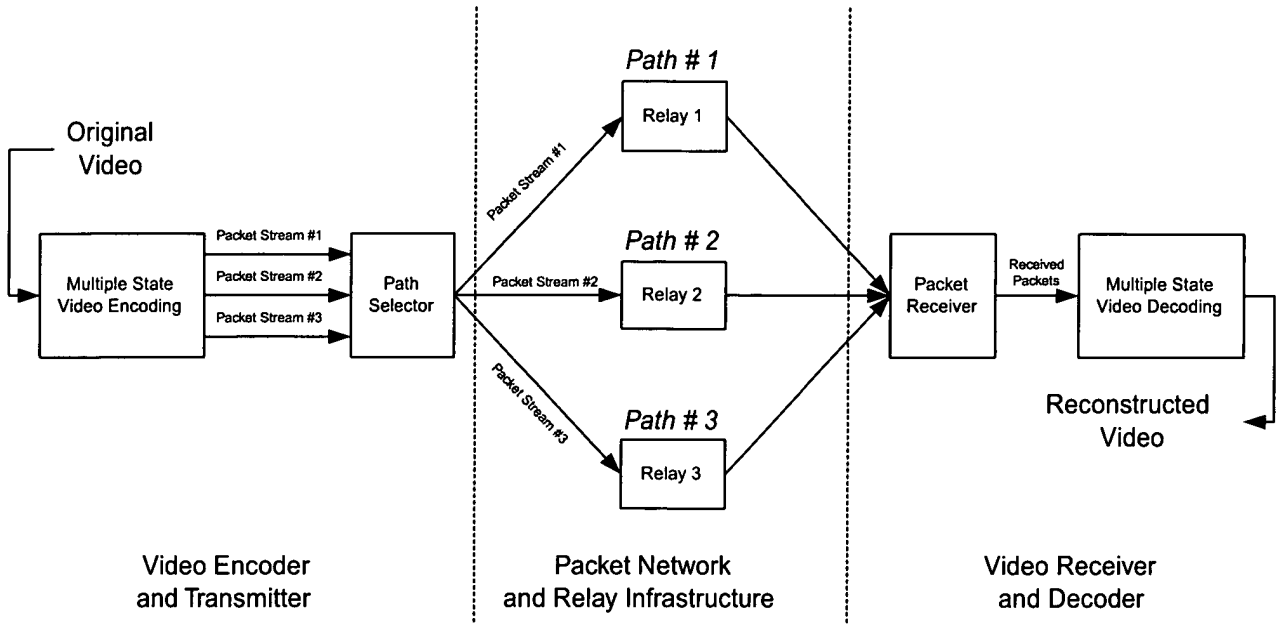

Figure 5. An example of the proposed system for reliably communicating compressed video over a lossy packet network, such as the Internet. The original video is encoded with multiple state encoding with three states into three independently decodable packet streams. Each packet stream is explicitly sent over a different path through the network by sending each stream through a different relay which forwards the packets to the final destination. Each path may be explicitly created through the use of one or more relays (only one relay is used for each path in the figure). The receiver receives the multiple streams and performs multiple state decoding to reconstruct the video for display. If there are no packet losses then the decoding is straightforward. If there are packet losses, the multiple state decoder performs state recovery to recover the corrupted stream based on the correctly received packets.

(IID) processes. If the (temporary) outage probability of each path is P_{outage}, then the outage probability of communication over multiple paths goes as $(P_{outage})^N$ where N is the number of paths. For example, if the probability of outage of a single path is .01 (e.g. 1 outage every 100 secs) and there are two paths with equal outage probabilities and which undergo independent outages, then the probability of outage using path diversity is .0001 (one outage every 2.8 hours). While this highly simplified example assumed that there were independent outages for each path and that there exists no single point of failure, the reduction in outage probability can still be quite dramatic even if these assumptions are not quite true. A reduction in the probability of an outage is very beneficial in a video communication environment, as well as in other applications which either have a delay constraint or for which retransmission is not possible. Furthermore, this property may also be beneficial for other applications which do not have a delay constraint and typically use retransmits, such as conventional web browsing, FTP, or telnet. An outage adversely affects all applications that require communication across the network, so the ability to reduce the probability of an outage appears to be beneficial to all applications that require this capability.

The primary motivation for this work has been for feedback-free video communication applications over lossy packet networks where either retransmissions are not possible (e.g. broadcast or multicast) or where the delay of retransmission is not acceptable (e.g. real-time video communications). However, it is important to note that the benefits described above may be useful in more general circumstances. For example, reducing the channel variance makes FEC-based systems more efficient and effective, and FEC may be useful in a variety of applications. Similarly, reducing the probability of a communication outage has potential benefits in a variety of applications.

6. PROPOSED MULTIPLE STATE ENCODING AND PATH DIVERSITY SYSTEM

The multiple state encoding and path diversity approaches presented in the previous sections can be combined to create a system for reliably communicating video over a lossy packet network (such as the Internet) by:

1. Applying the multiple state coding approach to encode the video into multiple independently decodable packet streams
2. Explicitly transmit each packet stream over a different path over the packet network through the use of a relay infrastructure
3. Receiving the multiple packet streams (potentially with some packet losses) at the receiver
4. Applying multiple state decoding to
   (a) Decode the video
   (b) Recover from any packet losses by performing state recovery based on the correctly received packets

An example of the proposed system is shown in Figure 5. The multiple state encoding subsystem codes the video into multiple independently decodable packet streams. In this example the video is coded into three independently decodable streams in an appropriate manner so that any one of these streams can be used by itself to reconstruct the video at 1/3 of the original frame rate. The multiple state encoding and decoding subsystem has the property that as long as any one stream is (largely) received correctly, the data in that stream can be used to produce a reduced-frame rate version of the video, and with the correctly received portions of the damaged streams can potentially recover the full frame rate of the video. The success of this subsystem requires that at least one of the streams is received correctly. The subsystem that explicitly sends the different packet streams over different paths in the network increases the probability that at least one stream is received correctly. In typical packet networks congestion is intermittent and varies from one path to another. If the paths are chosen such that they have independent losses (partially achieved through the strategic placement of the relay nodes) then the probability of congestion can be significantly reduced. The key point is that as long as any of the three paths is largely error-free then the multiple state decoding can provide usable video and attempt to recover the full video quality. By explicitly sending the appropriately compressed video over the separate paths, we are maximizing the probability that at least one packet stream is received error-free. Ideally, these separate paths are appropriately selected based on knowledge of their characteristics (probability of congestion and outages, dependence and independence of paths, etc.) to maximize the probability that at least one packet stream is received correctly.

Two important system design questions are (1) how many states should be used for encoding and (2) how many paths should be used for transmission. In multiple state encoding the ability to provide improved recovery arises as soon as two states are used. On the other hand, the compression efficiency decreases as the number of states increases since the frames which are predicted are spaced further apart in time and therefore motion-compensated prediction is generally less effective. Therefore, the use of two states (two independently decodable streams) appears to be a good compromise between providing improved recovery from errors and maintaining good compression efficiency. An example of the tradeoff between compression efficiency and recovery from errors is examined in the next section.

The question of how many paths should be used to achieve appropriate path diversity is more complex. Since the probability of outage decreases exponentially with the number of paths (assuming an IID model for each path) the more paths the better. On the other hand, the variance decreases with $\frac{1}{\sqrt{N}}$, where N is the number of paths. Therefore, while more paths are beneficial for reducing the variance, there are diminishing returns with increasing number of paths. If the above models are accurate, and if the complexity of the communication is related to the number of paths chosen, the above suggests to choose the minimum number of paths that would achieve the desired probability of outage and variance. As an example, the choice of two paths yields a relatively simple solution that provides a significant reduction in the probability of outage and also reduces the variance by $\frac{1}{\sqrt{2}}$. The use of two paths also straightforwardly couples to multiple state encoding using two states, as each independently decodable stream can be sent over a separate path (note that the number of states and number of paths can be different).

7. EXPERIMENTAL RESULTS

This section examines the effectiveness of the proposed multiple state encoding and path diverity system. As proposed in the previous section, multiple state encoding is performed with two states, producing two independently decodable streams, and two paths are selected for path diversity, such that each stream is sent over a different path. These tests assume the existence of a path diversity system, based on either of the two architectures that we proposed, providing two paths with independent losses. The effectiveness of the proposed video communication system was then examined in the context of implemented multiple state encoding and decoding and a model of an ideal path diversity system which provides two paths with independent losses.

The effectiveness of the proposed multiple state video coding and path diversity system is examined for communicating the Bus (240 × 352 pixels/f, 30 f/s) and Foreman (144 × 176 pixels/f, 30 f/s) sequences over a lossy packet network. The multiple state video coding algorithm was based on a software codec similar to MPEG-4 Version 2 (using NEWPRED) and H.263 Version 2 (using RPS) algorithm.

In the proposed system, each sequence was coded into two streams (containing the even and odd frames) at 15 f/s each and at a constant quality (PSNR) of 29.5 dB for Bus and 31.9 dB for Foreman. Coding the video into these two streams requires approximately 53 kbits/P-frame for Bus and 4.45 kbits/P-frame for Foreman. As a point of comparison, these bit rates are approximately 12 % and 20 % larger for Bus and Foreman (at the same quality) than for the corresponding single-state system designed for communication over an error-free channel (47.2 and 3.7 kbits/P-frame, respectively). The conventional single-state coding for an error-free channel uses a minimum of intra coding, and therefore is highly sensitive to errors and error propagation. To make an appropriate comparison, both the conventional single-state coding and the proposed two-state system were used to code Bus and Foreman at the same quality and the same bitrate (57.8 and 4.72 kbits/P-frame), where the extra bits were devoted to additional intra coding to reduce the potential for error propagation. For simplicity, in the following tests we assume that any loss leads to the loss/corruption of one entire frame or multiple entire frames. This assumption is appropriate for the Foreman sequence, where an entire P-frame fits within a single packet, however it is somewhat inappropriate for the Bus sequence, where a P-frame typically requires about 5 packets.

The effectiveness of a number of state-recovery methods was examined. We developed a state recovery method, MCinterp, which estimated the lost frame by performing a motion-compensated interpolation between the previous and future correctly received frames in the other stream. The motion between the two correctly received frames was first estimated using a phase-correlation motion estimation algorithm.[26] Specifically, corresponding 16 × 16-pixel blocks in each frame were windowed with a 2-D cosine window with square support (48 × 48-pixels for Foreman and 64 × 64-pixels for Bus), and then the cross-correlation was computed between them in the 2-D spatial Fourier domain. The peaks in the correlation surface were selected as candidate motion vectors for each block, and these candidate vectors were then used to generate a dense motion field (one motion vector per pixel). The dense motion field was used to estimate the lost frame as the motion-compensated average of the correctly received surrounding frames. While basic occlusion detection and processing was performed, further improvements can easily be achieved.

In addition to the sophisticated state-recovery method described above, four low-complexity methods were also examined. InplaceMC avoids computing motion by using the coded motion vectors between the previous and next odd frames, scaling them by 1/2 and applying them inplace to the previous odd frame to estimate the current even frame. The other methods examined were averaging the previous and next odd frames, using the previous odd frame, and using the previous even frame.

The effectiveness of recovering a lost frame using each of the recovery methods is illustrated in Figure 6. In these tests we assume that the even sequence has an error which corrupts one entire frame while the odd sequence is received correctly, therefore the odd frames and correctly received even frames are used to estimate the lost even frame. The horizontal axis specifies the even frame that was lost and the various plots illustrate the accuracy for which that lost frame was estimated using each method. This figure illustrates the variability in the recovery accuracy for different frames within the same sequence, i.e. most frames of the bus sequence can be recovered with approximately the same accuracy for a given recovery method, while the recovery accuracy for foreman varies significantly depending on the specific frame that is lost. Note that the PSNRs in Figure 6 are with respect to the lost coded frame, and not to the original frame, since the goal is to recover the coded frame. The PSNR of the MC-prediction of the lost even frame is also plotted, "PSNR MC-P even", to provide an indication of how well the lost frame can be estimated. For the bus sequence MCinterp performs best with an average PSNR of 24.2 dB, followed by InplaceMC (20.6 dB) and

simple frame averaging (18.1 dB). MCinterp also performs best for the Foreman sequence (32.4 dB), but it is closely followed by frame averaging (32.0 dB), and then InplaceMC (29.9 dB).

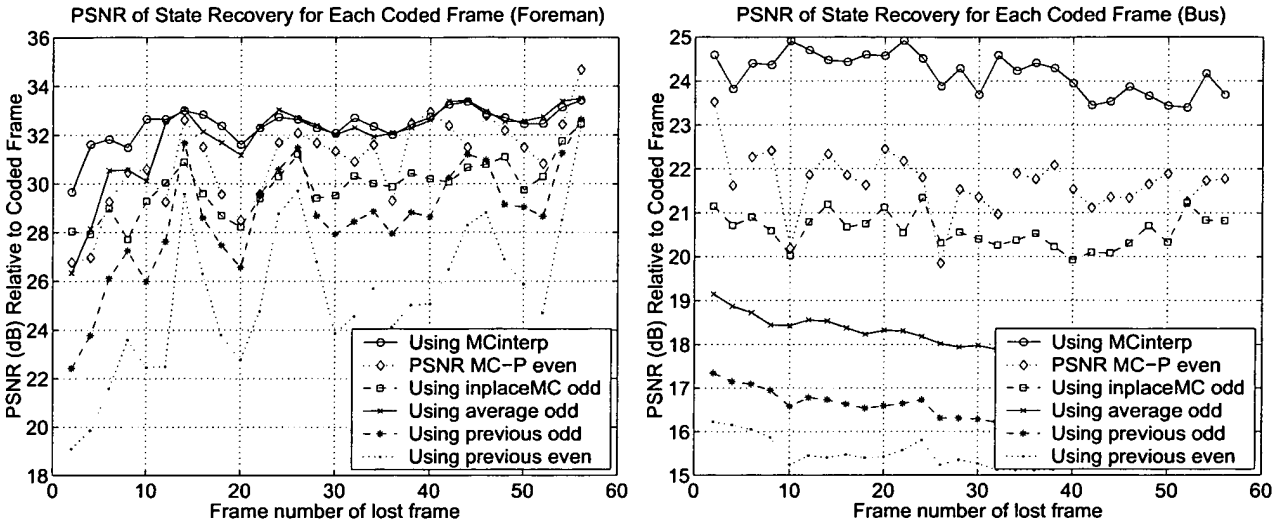

Figure 6. Accuracy of state-recovery as a function of lost frame for the Foreman (left) and Bus (right) sequences.

A number of experiments were performed to examine the effectiveness of the proposed multiple state encoding and path diversity system versus the conventional single state system with a single path when both are operating at the same PSNR and the same bit rate. Three different types of losses were examined: (1) single packet loss, which corresponds to the loss of a single entire frame (frame 10), (2) burst loss of 100 ms duration, which corresponds to the loss of three frames (starting at frame 10), (3) two burst losses of 100 ms duration, spaced apart by 2/3 sec, which corresponds to the loss of three frames in two locations spaced apart by 2/3 sec (starting at frame 10 and frame 30). In the proposed multiple state coding and path diversity system, we assume independent losses on each path. Specifically, in case (2) three frames are lost in the even sequence, and in case (3) three frames are lost in the even sequence starting at frame 10, and three frames are lost in the odd sequence starting at frame 31. In these tests, the proposed multiple state encoding applies MCinterp to perform the state recovery, and the single-state approach estimates the lost frame as the last correctly decoded frame.

Figure 7 illustrates the performance of each system under the different losses. With the Foreman sequence and the proposed system, a single frame loss leads to approximately a 1.5 dB loss in quality, while for the conventional system the loss is about 7 dB. With the Bus sequence and the proposed system, a single frame loss leads to approximately a 7 dB loss in quality, while for the conventional system the loss is almost 14 dB. While the single-state approach has a higher percentage of intra-coded macroblocks, allowing it to converge slightly faster to the point of complete recovery (assuming no losses during this period), it is significantly more vulnerable to losses.

The 100 ms burst loss (loss of three consecutive frames) has minimal affect on the proposed system beyond that of a single loss, however it leads to an additional 5 dB loss for the conventional system for Foreman, and 2 dB for Bus. This example illustrates that the proposed multiple state encoding and path diversity system is largely immune to the duration of the loss in one channel, as long as the other channel is correctly received. This contrasts to the conventional approach, where longer durations of loss lead to greater reductions in quality. In addition, the multiple state decoder has the capability to monitor the quality of the recovered video, and decide whether or not it is appropriate for display. For example, if it is deemed inappropriate, the decoder can choose to simply display the clean subsequence, leading to a temporary reduction in the frame rate but without other artifacts.

The two 100 ms burst losses illustrate the effect of a significant loss on a system while it is still recovering from a previous loss. In the proposed system, as long as both paths are not lost simultaneously, at least one of the states will be (largely) uncorrupted, and can assist in recovering the other. The two separate states can in effect bootstrap off of each other, as long as they are not both simultaneously corrupted.

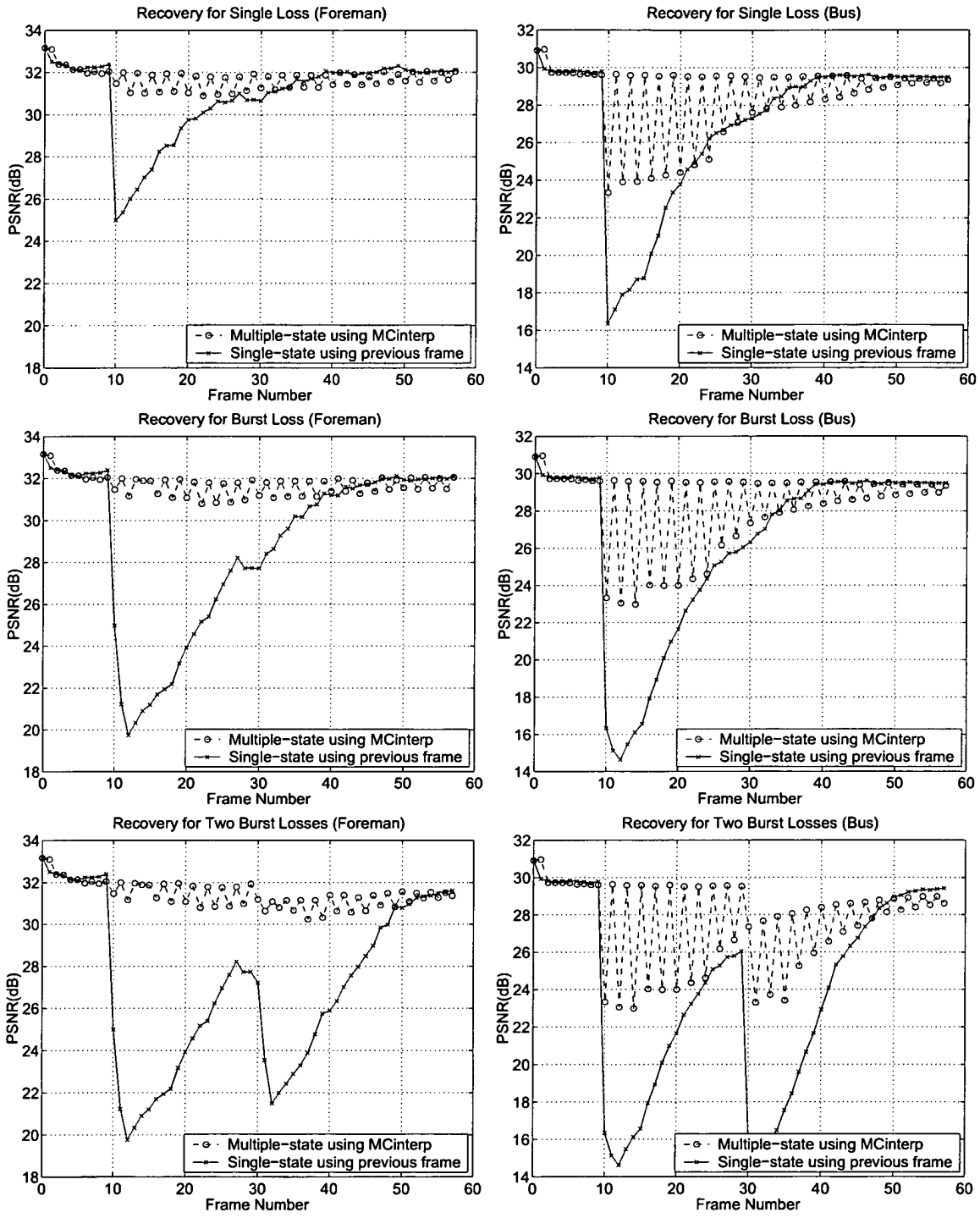

Figure 7. Recovered video quality for the Foreman (left column) and Bus (right column) sequences when one entire frame is lost (top row), burst loss of 100 ms duration corresponding to the loss of three consecutive frames (middle row), and two burst losses of 100 ms duration, spaced apart by 2/3 seconds, corresponding to the loss of three consecutive frames in two separate locations spaced apart by 2/3 seconds.

8. SUMMARY

This paper presents a system for reliable video communication over lossy packet networks such as the Internet. The proposed system is composed of two jointly designed subsystems: (1) multiple state video encoding and decoding, and (2) path diversity transmission system. The video is coded into multiple independently decodable streams where each stream is explicitly sent over a different path ideally chosen to have independent losses. If one stream is lost than the other correctly received streams are used to perform state recovery and recover the corrupted stream. The novelty of multiple state encoding is that it provides and uses bidirectional (previous and future) information to enable improved state recovery for the corrupted stream, as compared to conventional approaches which only have access to previous information. In the event of a loss, the characteristics of the video quality for a MC-prediction based coder is such that there is a sudden quality loss followed by a gradual recovery. The speed of recovery depends largely on the amount of intra coding. The loss in quality depends on the decoder's ability to estimate the lost information. Our experimental results show that when burst loss afflicts the Foreman and Bus sequences originally coded at 31.9 and 29.5 dB, respectively, the conventional approach results in initial PSNR drops of 12 and 15 dB, while our proposed multiple state approach results in initial PSNR drops of only 1.5 and 7 dB.

The proposed path diversity system explicitly sends different subsets of packets over different paths, as opposed to the default scenario where the stream of packets proceeds along a single path. As a result, the probability that all of the multiple streams that are transmitted on different paths are simultaneously congested and have losses is much less than the probability that a single path is congested. We propose two architectures for providing path diversity by explicitly sending a stream of packets through multiple paths. One of these approaches is based on the use of relays; while this approach requires some infrastructure (the relays), it does not require other access to the network (e.g. routing tables). This relay infrastructure corresponds to an application-specific overlay network on top of the conventional Internet. Its goal is to route traffic through semi-intelligent nodes at strategic locations in the Internet, thereby providing a service of improved reliability while leveraging the infrastructure of the Internet.

Multiple state video coding and path diversity are useful even if used separately. For example, multiple state video coding can provide improved reliability even when sent over a single path. In addition, it does not require a back-channel and therefore can be applied in a wide variety of applications (e.g. broadcast or multicast), and it has the attractive property that it can be applied as a standard-compatible enhancement within MPEG-4 Version 2 (with NEWPRED) and H.263 Version 2 (with RPS)[¶]. Path diversity provides a number of benefits including (1) a reduced variability in communication quality as compared to an individual path, (2) burst packet losses are converted to isolated packet losses, and (3) the probability of an outage is greatly reduced. Therefore path diversity may be beneficial for more general packet-based communication system design, as it provides an improved virtual channel and simplifies system design, e.g. FEC-design. When used together, multiple state video coding and path diversity complement, and also to a certain extent enhance, each other's capabilities. Multiple state video coding provides multiple independently decodable bitstreams, which the transmission system explicitly sends over different paths, and the transmission system provides the video decoder with a high probability that at least one of the streams will be received correctly at any point in time, thereby enabling the video decoder to perform state recovery to recover a corrupted stream. Based on the above discussion and experimental evidence, multiple state video coding and packet diversity appear to provide a promising direction for attacking the problem of reliable video communication over lossy packet networks.

[¶]Therefore any MPEG-4 Version 2 decoder can decode the resulting bitstream while an enhanced decoder designed to perform state recovery as presented in the paper can provide improved error recovery.

ACKNOWLEDGMENTS

The author would like to thank Susie Wee for discussions on video coding, Gregory Wornell for discussions on diversity, and Tina Wong for pointers to relevant networking literature.

REFERENCES

1. V. Vaishampayan and S. John, "Interframe balanced-multiple-description video compression," *ICIP* , Oct. 1999.
2. R. Talluri, "Error-resilient video coding in the ISO MPEG-4 standard," *IEEE Communications Magazine* , pp. 112–119, June 1998.
3. International Organization for Standardization, MPEG Committee, *Information Technology - Generic Coding of Audio-Visual Objects: Visual, Working Draft Version 2, N2553*, December 1998.
4. International Telecommunication Union, *Video Coding for Low Bit Rate Communication, ITU-T Draft Recommendation H.263 Version 2*, September 26, 1997.
5. N. Färber, B. Girod, and J. Villasenor, "Extension of ITU-T Recommendation H.324 for error-resilient video transmission," *IEEE Communications Magazine* , pp. 120–128, June 1998.
6. D. Redmill and N. Kingsbury, "The EREC: An error-resilient technique for coding variable-length blocks of data," *IEEE Transactions on Image Processing* **5**, pp. 565–574, April 1996.
7. S. Fukunaga, T. Nakai, and H. Inoue, "Error resilient video coding by dynamic replacing of reference pictures," *Proceedings of GLOBECOM* **3**, pp. 1503–8, November 1996.
8. E. Steinbach, N. Färber, and B. Girod, "Standard compatible extension of H.263 for robust video transmission in mobile environments," *IEEE Trans. on Circuits and Systems for Video Technology* , pp. 872–881, Dec. 1997.
9. B. Girod and N. Färber, "Feedback-based error control for mobile video transmission," *Proceedings of the IEEE* , pp. 1707–1723, October 1999.
10. S. Wenger, G. Knorr, J. Ott, and F. Kossentini, "Error resilience support in H.263+," *IEEE Transactions on Circuits and Systems for Video Technology* , pp. 867–877, November 1998.
11. M. Budagavi and J. Gibson, "Error propagation in motion compensated video over wireless channels," *Proceedings of the IEEE International Conference on Image Processing* , pp. 89–92, October 1997.
12. T. Wiegand, N. Färber, K. Stuhlmüller, and B. Girod, "Error-resilient video transmission using long-term memory motion-compensated prediction," *IEEE Journal Selected Areas in Commun.* , pp. 1050–62, June 2000.
13. M. Budagavi and J. Gibson, "Multiframe video coding for improved performance over wireless channels," *IEEE Transactions on Image Processing* , To appear.
14. A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, "Multiple description coding for video using motion compensated prediction," *IEEE Inter. Conf. Image Processing* , October 1999.
15. J. Apostolopoulos, "Error-resilient video compression via multiple state streams," *Proc. International Workshop on Very Low Bitrate Video Coding (VLBV'99)* , pp. 168–171, October 1999.
16. W. Tan and A. Zakhor, "Real-time internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Trans. Multimedia* , pp. 172–186, June 1999.
17. D. Clark and D. Tennenhouse, "Architectural considerations for a new generation of protocals," *ACM SIGCOMM* , September 1990.
18. Y. Wang and Q. Zhu, "Error control and concealment for video communications: A review," *Proceedings of the IEEE* , pp. 974–997, May 1998.
19. A. Tanenbaum, *Computer Networks*, Prentice Hall PTR, New Jersey, 1996.
20. W. Tan and A. Zakhor, "Error control for video multicast using hierarchical FEC," *Proceedings of the IEEE International Conference on Image Processing* , pp. 401–405, October 1999.
21. "Special issue on 'real-time video over the Internet'," *Signal Processing: Image Communications* , Sept. 1999.
22. H. Poor and G. Wornell, *Wireless Communications: Signal Processing Perspectives*, Prentice Hall, N.J., 1998.
23. V. Paxson, "End-to-end internet packet dynamics," *Proc. of the ACM SIGCOMM* , pp. 139–152, Sept. 1997.
24. A. C. Savage, S., E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of internet path selection," *Proceedings of the ACM SIGCOMM* , October 1999.
25. J. Apostolopoulos and G. Wornell, "A system for enabling reliable communication over lossy packet networks via path diversity," *HP Internal Report, To be published externally* , 1999.
26. G. Thomas, "Television motion measurement for DATV and other applications," *Research Department Report, British Broadcasting Corporation* , September 1987.